Realising an Applied Gaming Eco-system

Research and Innovation Action

Grant agreement no.: 644187

D1.2 – [Asset Integration Methodology]

RAGE – WP1 – D1.2

Project Number	H2020-ICT-2014-1
Due Date	30 November 2018
Actual Date	24 November 2018
Document Author/s	PH, DG,SW, WW,WVV, KS, BM,IMO JG, TP, JC, MH, AP
Version	Final
Dissemination level	PU/ RE
Status	Final
Document approved by	

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 644187

Document Version Control			
Version	**Date**	**Change Made (and if appropriate reason for change)**	**Initials of Commentator(s) or Author(s)**
0.1	10/02/18	Outline Chapters/ Notes	PH
0.2	01/11/18	Integration Content Added	PH
0.3	11/11/18	Additional Content and Editing	PH
1.0	20.11/18	Major Edit	PH
2.0	26/11/18	Edit and additional content	PH
Final	27/11/18	Edit and additional content	PH

Document Change Commentator or Author		
Author Initials	**Name of Author**	**Institution**
PH	Paul Hollins	UOB
DG	Dai Griffiths	UOB
WvW	Wim Vander Vegt	OUNL
BM	Baltasar Manjon	UCM
KS	Kressen Stefanov	US
WW	Wim Westera	OUNL
SW	Scott Wilson	UOB
IMO	Ivan Martinez-Ortiz	UCM
JG	Jared Glass	PlayGen
JC	Jeremy Cooke	Gameware
TP	Thierry Paton	BIP
MH	Matthius Hemmje	FTK
AP	Andrew Pomazanskyi	Nurogames

Document Quality Control			
Version QA	**Date**	**Comments (and if appropriate reason for change)**	**Initials of QA Person**

Final	27/11/18	Approved	JG (Playgen)
Final	30/11/18	Approved	JJ (Utrecht University)

TABLE OF CONTENTS

Page

LIST OF FIGURES

EXECUTIVE SUMMARY

This deliverable is primarily written for the Executive Management Board (EMB) of the RAGE project and deals with the asset integration methodology employed in the RAGE project.

A considerable amount of the effort of the RAGE project has been directed towards addressing the technical challenges of asset integration and interoperability. All RAGE partners engaged in asset development adopted a pragmatic approach to integration and interoperability and where practical adopted the use of Open Standards and Specifications supported by a defined project architecture. Asset developers adopted a consistent approach to development languages.

The four RAGE Game development partners each included a number of assets within their development processes and in producing the RAGE pilot Games. Their experience of the asset-based development methodology is detailed in this deliverable.

Where potential gaps in the interoperability integration activity were identified, specifically in interoperability between the output of games data and Learning Management systems, the project undertook specific "proof of concept" work to address this.

Where proposed activity was identified by the development community or through consultation as being problematic or as a potential barrier to adoption, for example the case of a formal nomenclature, effort was directed to more productive activity.

This approach unearthed technical challenges, which were managed by the RAGE project Integration and interoperability group All issues were successfully resolved during the research and development phase of the project. Ultimately only minor integration problems occurred and these are documented within the Game development partners input into this deliverable.

The RAGE project's pragmatic approach to integration and interoperability has been very well received by those engaging with the project also serving to "cross the chasm" from both technical and philosophical perspectives and between academia and Industry. In doing so the project has achieved a number of the key primary objectives.

Conceptually, from a technical perspective, the RAGE asset-based approach to Applied Game development has been validated.

INTRODUCTION

This document reports on the approach and activities relating to integration and interoperability as specified in tasks 1.2 and 1.4 of the grant agreement. These are summarised as follows.

Task 1.2: Specifying the interoperability requirement of Assets

This task has two aspects, which are paraphrased as follows:
Firstly, it requires the project to comply to a set of existing technology standards and specifications for easy connection or integration with existing game platforms and LMS systems.
Secondly, it seeks to maximise the mutual interoperability of RAGE assets, by exchanging state data, either directly or through an intermediate agent.

The outcome of task 1.2 is a substantiated decision about the formats and standards to be used in linking assets together and integrating assets with existing game platforms.

Task 1.4: Linking assets with selected Games Platforms

This task focuses on facilitating the integration of assets in existing game platforms. More specifically, it deals with the data, special software components, objects to be rendered, and the interface for data exchange. Proof of concept integrations have been carried out with approachable games platforms, combining the use of existing interoperability specifications and a database of RAGE asset component extensions. This work has led to the development of a generalized methodology for the integration of RAGE assets in game platforms.

The first section deals with the outline approach of the project to integration and interoperability and to the use of standards and specifications in this respect. The broad approach to the use of standards can be defined as “pragmatic” and this philosophy has underpinned the strategy in dealing with the integration and interoperability challenges faced. Where practical open standards have been employed, the rationale behind their preferential use is detailed within the document. The pragmatic approach extends to recognising of the need, where they are necessary or add value, to use de-facto industry proprietary standards. This flexibility was critical to engaging the Game development community and to ensure the long-term sustainability of the project outcomes
The second part of the first section provides in depth detail of the RAGE component architecture and evaluation of the said architecture by development users. The evaluation concludes that both Asset and Game developers “appreciated” the architecture as being effective in both Asset and Game development.
The second section of the document presents the reflections of each of the Game and Asset development partners regarding their experience of both the RAGE approach to integration and interoperability and to the Asset integration work undertaken. These reflections highlight the problems and challenges of integration and interoperability and suggestions, if any, for further development. The reflections provide validation of the pragmatic approach in concluding that few “unresolvable” problems occurred during the development process.
The third section of the document discusses the project’s approach to asset descriptions and nomenclature.
The fourth section of the document provides detail on the specifications and standards considered and the rationale behind additional development work undertaken by the project. Section Five provides details of the contribution and impact of the project input into international Open standards development initiatives focussing specifically on the contribution of the xAPI (Experience Application Programming Interface) Serious (Applied) Game Profile developed by RAGE partner UCM. Section six provides detail of the integration challenges addressed in three scenarios when using Applied Games within in or in parallel to a Virtual Learning Environment (VLE) or Learning Management System (LMS). This involved the project undertaking an LTI - Moodle integration “proof of concept” work and highlights future development opportunities beyond the RAGE project.

Section 7 provides conclusions drawn from the work on this deliverable.

1. INTEGRATION APPROACH AND RATIONALE

In this section we describe how the Description of Work (DOW) has been addressed in the two related lines of project activity: integration and interoperability. The distinctions between the two activities are clarified, and an overview provided of the work carried out.

A RAGE project internal Integration and Interoperability working group was established, which included those partners with management responsibility for the oversight of integration and interoperability, in particular those involved in WP1 and representatives of other associated Work Packages, namely WP2, WP3 and WP4.

The group met periodically as required during the duration of the project with the aim of to ensure that integration and interoperability issues were shared and addressed through collaboration across the project. A collaborative registry on Google docs, with open access to RAGE parties, “Asset Interoperability issues” was maintained for the duration of the project. Details of issues were documented and shared together with mitigating action and resolution. Details of any prospective interoperability issues anticipated with the assets were listed in order that they may be considered in future planned development work.

Whilst Interoperability is a prerequisite within the Technology Enhanced Learning (TEL) domain, Interoperability and (open) standardisation are not yet perceived as a necessity or indeed desirable within much of the game development Industry. There remains a long-standing debate between those who support and develop using open standards and those who maintain proprietary standards and software to maintain what they perceive as a “competitive advantage”. This was a constant theme of discussions with developers over the duration of the project.

The counter argument is that (open) interoperability standards if adopted broadly across industry, would serve to grow the overall market size through ease of adoption and use, which is entirely consistent with the RAGE project objectives.

This dichotomy presented a challenge to the RAGE project and in achieving the project’s overarching objective of “stimulating Applied Games development in the European Union” and in turn contribute to overall growth of the market and demand for Applied Games.

One of the challenges facing the RAGE project and a potential metric for success is in informing the development partners within and outside the project of the benefits of adopting Open standards and a flexible open approach to development, one that, in line with the objectives of the project, stimulates commercial development of applied games and contributes to create a market for Applied games in different industries.

For example, game producers that employ learning analytics software are often doing so with proprietary systems and proprietary data formats considering this, as previously mentioned, as a competitive market advantage. RAGE has raised the awareness about the benefits of using different standards to enlarge the applied game industry. This was done in a pragmatic way, driven by the demand, and the project can now provide exemplars and reference models to demonstrate the benefits of assets underpinned by open standards.

Where possible and practical to do so all outputs including the assets developed by the RAGE project are consistent with European Community (EC) policy and are interoperable and incorporate the use of “open” specifications and standards.

The project’s pragmatic approach to the use of open specifications and standards adopted can be described as a broadly “lite” touch. The project initially considered adopting a three level “mandation” status framework of specifications and standards but as work evolved there was little enthusiasm for a more formal regulatory framework which was identified as having little benefit.

Developers embraced the pragmatic approach to integration and interoperability through adopting standards and specifications as required and to address specific issues. In games there are many

different aspects and at different levels there are candidates for standardisation such as interoperability of complete games across different platforms, interoperability of game assets, game assets descriptions or data formats for exchanging data about the games.

We recognised the interdependency of work packages in informing our approach to standards, allowing and supporting developers in the creation of the RAGE assets but overall an approach that stimulates innovation as opposed to hindering development with over ambitious unrealistic expectations. We recognise there are different levels of desired interoperability underpinned by standards in the project. There are specific (internal) requirements determined by the asset metadata description.

In addition, we considered ...

- There are less specific generic standards such as web standards that will be incorporated within the developed assets
- There are specific standards, largely in the technology Enhanced Learning (TEL) domain, that enable data exchange (or partial interoperability)
- There are specific standards that will enable full interoperability of the assets with multiple platforms and systems.
- There will be proprietary (platform specific standards) enabling functionality on proprietary platforms.

As previously stated we recognised that developers may be required to apply proprietary or industry “de facto” standards in striving for innovation, the Unity components being a case in point.

1.1 Interoperability of Games across Platforms

Due to the variety and diversity of game platforms (e.g. consoles, computers, tablets, smartphones), there is not a standardisation or any initiative covering the interoperability of games across all kind of platforms. This interoperability is even more complex due to the requirement imposed by some vendors about the games that can be included in their markets (e.g. Apple) or game platforms (e.g. Microsoft, Playstation, Nintendo).

As reflected in the initial study of development technologies undertaken by RAGE this aspect was covered by using cross-development environments, either commercial (e.g. Unity3D) or proprietary (e.g. C++), that are able to export the game for the required platforms.

To deal with the interoperability across platforms some developers used HTML5 technology as a means to deploy their games in different platforms (e.g. computer, Android) even if it did not address the (proprietary) console market.

The RAGE project purpose is not focussed on developing new standards or specifications when existing standards and specifications are available to address challenges and issues, although the project has contributed an API to an international standards initiative details of which are provided later in this document. The fact that the project has worked closely with key international initiatives of Assisted Distributed Learning (ADL) and IMS GLC was a very positive outcome.

1.2 The RAGE component architecture

This section details the rationale behind the design of the RAGE component architecture and results of an evaluation using the Technology Acceptance Model (TAM) undertaken by the project Asset developers.

The RAGE component architecture distinguishes between server-side components and client-side components. Remote communications of server-side components with centralised applications are based on a service-oriented architecture (SOA) using the HTTP-protocol (e.g., REST), which offers platform-independence and interoperability among heterogeneous technologies. Client-side components, however, which need to be integrated into client-machine applications (e.g. game engines), are likely to suffer from incompatibilities.

To avoid such client-side interferences as much as possible the RAGE client-side asset architecture was designed to avoid dependencies of external software frameworks as much as possible. This activity served to demonstrate that the RAGE compliant components facilitate easy integration and portability.

Integration is designed in such way that much of the integration code needed for a RAGE component to perform its tasks can be re-used (viz. storage of data and interaction with REST services). The Bridge code is key to the architecture, this provides for interfaces that fully shield the component's implementation details. Just the component´s public API (delivering the component's core functionality) is the only component specific integration code that cannot be reused.

Portability extends across multiple dimensions such as development environments, game engines, programming languages, mobile and desktop target platforms and underlying operating systems.

Both technical integration and portability of the architecture have already been assessed Van der Vogt et al (2016). Further details of the architecture are provided in Deliverables D1.1 and D1.4.

Formatted: Dutch (Netherlands)

Likewise, the specific (pedagogical) functionalities being delivered by RAGE components have been evaluated separately by integrating components into serious games and evaluating those games with end-users (Deliverable D8.3).

In addition to those evaluations, a separate architecture acceptance study, particularly focused on the client-side part of the architecture was carried out, which will be briefly reported below.

1.2.1 Architecture Acceptance Evaluation

The acceptance of the RAGE client-side asset architecture (RCSAA) by both component developers and game developers was investigated at the start of 2018 after most of the component development work and game coding was completed and subjects had sufficient experience with the architecture. The acceptance of the RCSAA by programmers, both component creators and component users (i.e. game developers) of the RCSAA was validated amongst a limited set of the portability dimensions: C# programming language and Unity3D game development platform, predominantly being used for the RAGE games.

The evaluation was performed with two questionnaires (for component developers and component users, respectively) that used a shared set of questions, be it with wording adjusted to the different viewpoints (component creation vs usage). Both questionnaires were built around TAM (Technology Acceptance Model) as the main instrument to assess both perceived usefulness and ease of use, which are considered key indicators for acceptance. TAM was complemented with some basic demographics, a programming experience self-estimation, and a set of questions related to architecture usage (architectural features, communication modes and bridge interfaces used).

The TAM questions consisted of a set of 6 Likert scale items measuring perceived usefulness and another set of 6 items for measuring ease of use. Furthermore, the questionnaires for game developers included questions on attitude towards use of 3rd party code and the code's origin (research institutes). Although strictly speaking this is outside the scope of the architecture, as it considers the core code implementing the pedagogical functionality, it was included as a potentially influencing factor for the architecture acceptance.

1.2.2 Results

Out of the eighteen component developers that responded, five claimed to have worked on server-side systems and skipped the client-oriented TAM questions. They were thus excluded from TAM analysis.
Five game developers responded, representing all game studios participating in RAGE. The developers reported to be involved in development of all 7 RAGE games.

The programmer's age date shows a bimodal distribution with two peaks, one below 25 and the other around 40 years for both groups.

The self-assessment of programming skills shows relatively high scores for all developers with the exception of the TypeScript language (which is a relatively new programming language introduced to add compile time type checking and object-oriented constructs to JavaScript). Java skills where rated higher amongst component developers which may be attributed to the development of high-performance server-side systems in Java.

The TAM questionnaire showed a good to excellent internal consistency (Cronbach's alpha) amongst both target groups.

The TAM results for component developers showed a 0.55 mean score for perceived usefulness and a 0.64 mean score for ease of use on the [0,1] interval scale, both with a standard error of 0.05, indicating an overall positive attitude towards using the architecture. Component developers indicated that the architecture was easy to understand and helped them to be more effective in creating and delivering components.

Game developers' scores are slightly lower: 0.53 for perceived usefulness and 0.58 for ease of use, with standard errors of 0.08 and 0.12, respectively, indicating the architecture as moderately usable. However, the scores where negatively biased by a single (outlier) developer assigning much lower scores compared to other developers. Due to the small number of respondents the influence was quite substantial. Removing this single outlier results in a perceived usefulness score of 0.62 and ease of use of 0.64, with standard errors of 0.10 resp. 0.06, indication a positive attitude towards the architecture.

Additional comments, however, reveal a preference to the traditional direct integration being used. Nevertheless, four out of the five game developers indicated an intention to keep using the RAGE architecture, while the fifth game developer indicated that their use would be dependent on the pedagogical functionality delivered.

Use of the different communication modes allowed by the architecture is decided upon by component developers during component design. As expected the communication from game to component (i.e. API) and vice versa (e.g. data storage, settings) where used most. Component to web service was the third most used. Communication between components and the publish/subscribe broadcasting were the least used. For the top 3 communication modes, the game developer has to supply the actual implementation using bridge interfaces. Broadcasting involves minimal coding in order to subscribe to events.

Overall, the practicability of the architecture was validated by both component developers and (most) of the game developers involved in the study.

2. CASE STUDIES

This section presents reflective summaries provided by each of the four RAGE Game Development partners; Playgen, Gameware Europe, BIP Media and Nurogames. They describe the use of assets within the RAGE case study pilot games where integration and interoperability or exchange of data has been achieved. The summaries highlight some of the specific challenges faced and where problems have emerged in achieving successful integration or interoperability.

The feedback and findings were used in a formative way to inform the developers and owners of assets in order to foster continued discussion and development.

The second part of this section provides details of work undertaken to contribute to formal specifications and standards bodies. Primarily this concerns the serious games profile for xAPI which has been developed by partner UCM for the project.

2.1 Case Study 1: Playgen

Playgen developed two of the RAGE game case studies; Sports Team Manager and Space Modules.

Both the Sports Team Manager and Space Modules Inc games were designed as soft skills training and as such simulating realistic and socially intelligent human interactions was imperative to both.

Players are presented with scenarios where they would have to problem solve while considering the effect and impact of their choices on the in-game character moods. To achieve this in both cases the FAtiMA toolkit (an emotion engine for Artificial Intelligent (AI) characters) was integrated. The FAtiMA toolkit is one of the most extensive assets. It is presented here as the core of the Playgen case study.

The documentation provided in the FAtiMA software was auto-generated and extensive but due to the nature of auto-generated documentation is has two main drawbacks:

1. As documentation is generated for all marked code, it can get quite verbose in non-critical areas resulting in developers having to sift through somewhat unnecessary documentation.
2. Because the focus is spread across large areas of code, areas that may be more important lose emphasis and don't necessarily get the detail they require.

A Unity demo was also provided by the asset developers. Using the Unity demo in conjunction with the available documentation proved to be a successful strategy in investigating various areas of functionality. Using this combined approach, game developers were able to determine how any interest areas of functionality could be used and should be integrated.

The performance of the FAtiMA toolkit didn't break the immersive game experience. Level loading at the start of the game and in between gameplay scenes is common in games and acceptable as these load times do not take place during periods of expected player action. Saving gameplay data however can and often needs to take place during a gameplay scene, especially in Sports Team Manager where a gameplay session can span a relatively long time.

For Space Modules Inc the solution was relatively straight forward as the play session was short enough to simply save when the session ended.

Given good access to the asset developers throughout the integration process we were able to integrate this asset into our first and then second game relatively easily. It was a relatively stable asset to start with but any of the minor bugs we reported were quickly resolved making the

integration process a relatively painless one. The conscious focus of the Playgen case study and feedback is on the integration of the FatiMa toolkit. This is due to the complexity of the tool and integration challenges faced,

There were no significant problems in integrating the RAGE assets into the Pilot games.

2.2 Case Study2: Gameware Europe

Gameware developed two RAGE game case studies Hatch and Interview Skills for Police Officers (ISPO)

Gameware used RAGE assets in each use-case and the comment is not specific to each game but applicable to both.

Server-Side Interaction/Storage and Analytics

The setup of these components requires the developer to operate their own or gain access to their own client's LINUX server, if one is available. The server needs setting up as per the instructions provided with the components.

Accepting that the pilot study Hull College UK (HCUK) was capable of setup, this process is more than likely to be beyond the scope of smaller software developers who are not familiar with server setup. If developers are prepared for any technical issues, they may be unlikely to have factored in the additional development cost involved in incorporating this solution during any costing phase, perhaps assuming either the component package would be provided and supported remotely (i.e. by the supplier) or based on Microsoft Windows Server. In our view, MS Server is far more likely to be in use by the smaller scale developers at who the RAGE eco-system is targeted.

For more rapid integration, these assets would benefit from wrapper classes which use simple functions and enumerations, such as:
Analytics.DoButtonClick("ok"),
Analytics.DoStartNewGame(),
Analytics.DoEndGame().

Readerbench - sentiment analysis on texts (and other Readerbench assets)

The documentation provided with the Readerbench components assumes the developer is familiar with server setup.

The web interface worked well. Faster integration could be achieved, by providing a C# wrapper, which would provide a class with simple property members for things such as LDA or LSA options, etc.
The actual weights of sentiments returned in general usage did not provide enough clarity to be used *per se*. To provide for this, we were required to implement an additional layer of processing to translate the sentiment weights returned to values more useful to the Hatch application.

Processing of results from the asset could be improved (in terms of ease of use) if the asset presented a wrapper which completed the JSON parsing for you and provides multiple simple methods which can be used to infer meaning from the results.

A wrapper could provide simple functions such as GetMajorSentiment for a phrase, which automatically parses the JSON, determines the returned most influential valence etc., or it could provide functions such as IsMostlyHappy, Is Negative, Issa etc.

Speech I/O - Text to speech

Overall, the text to speech engine worked very well in the Portuguese language. It was also straightforward to integrate. The asset would benefit from further development in placing pauses and improving inflection in the speech.

Similarly, if a phrase is sent which includes periods, it is more useful if the component returns a long audio file with pauses, than a series of audio files, as the application has to either setup a script to play these files one after the other, or concatenate them into a single audio file.

Additionally, there would be significant benefit from automatically dealing with semicolons (small pauses), and slang or vernacular speech such as “hmm”. As things stand, the calling application needs to process these instances itself.

Speech to Text

Again, this worked well, but would benefit from being real-time to allow an application using it to “highlight” suggested text, whilst the user is talking. This is likely to require the asset to be provided as a library and not a service.

Adaptation and Assessment (TwoA)

This asset was simple to integrate, particularly after the enhancements were added, to allow the asset to give adaptation values more readily and requiring less user input. In practice, its functionality was quite basic and could readily be included by a developer on a basis closely aligned with their own application needs.

The domain model asset needs to have documentation, which includes a full set of data required to use the domain model asset. This asset may also benefit from a more succinct documentation.

The game storage client-side asset is easy to integrate, but would benefit from additional functionality for larger data sets.

Overall our experience of integrating RAGE assets did not present any significant problems but the Asset integration experience would be improved significantly with the enhancements detailed in above.

2.3 Case Study 3: BIP Media

BIP Media developed the RAGE game case study Jobquest.

Challenges were experienced with some of the features and results/outputs provided by the RAGE components. Working closely with the Asset/components developers proved very useful in resolving these challenges .

Virtual Human Controller

This component is able to read and interpret BML Files. The BML language is used in dialogue and it simplifies the addition of facial expressions, emotions and other kind of animations.
For Job Quest, we used the Communication Scenario Editor, another RAGE component, for an easy editing of dialogues by the Randstad experts. This Communication Scenario Editor produces BML and is usable online, on a web site.
But the original Virtual Human, the BML parser, was also online.
BiP media had to re-engineer it for Unity which did prove challenging.

Real-time Emotion Detection

The demonstration in C++ gave good results but when integrated in Unity the “values” results were corrupted.

Real-Time Arousal Detection Using Galvanic Skin Response

This component uses a Windows service, so did not require any integration, but it was extremely challenging to run this on our external client and Randstad’s personal computers. After the calibration period it was challenging to interpret the results and output of the component.

ReaderBench - Semantic Models and Topic Mining

We worked very closely with the asset provider to improve functionality. Randstad furnished hundreds of CV as examples and the result was a unique functionality for the Job Quest game.

Interoperability

Some of the RAGE components can be integrated into Unity whilst some are online and require Internet connection, others are Windows services and require special Administrator rights to function fully.

The internal RAGE partner Randstad experienced challenges brought about by their internal security and firewalls. On a standard Randstad PC, the user has no access rights to install new applications, the applications have no rights to call another one and Internet access is strictly limited to certain ports, for example a new Windows service is not allowed to run, etc. This resulted in Randstad running the demonstration pilots outside their own firewall.

BIP did not encounter any significant difficulties concerned with integration or interoperability.

2.4 Case Study 4: Nurogames

Nurogames developed the RAGE game case study The WaterCooler Game.

Nurogames aim was to develop a game the focus of which was to improve users interpersonal and social skills in simulating various scenarios in the gaming environment which included various conversations between the Non-Playing Characters (NPC), work and conflict management supported by the behaviour and value-based questions and multiple-choice answers. In order to support reaching objectives, understand the outcomes and learning context of the game, the following RAGE components were evaluated and integrated into the game, the aim of which was to speed up the process of the development and where relevant provide an added value to the game.

Server-side Authorization and Authentication Asset

The integration and use of this particular component was straight-forward and easy to use by third party developers, the focus of which is to handle user accounts creation and authentication to manage various functionalities of the future game. Furthermore, it provides a single-sign-on capability to further components that are integrated within the game. No issues were detected whilst integrating or during the operation. The documentation is well written reducing any potential future questions from the third-party developers. The component is implemented as a node.js application offering the passport and the http-proxy node.js modules.

Server-Side Interaction/Storage and Analytics

The added value of the component is quite evident and for the most part of it can be one of the integral components for the game development studios. It provides a server that collects and analyses relevant data from the tracking clients. One of the prerequisites of this component is that the authorization and authentication components are required and the set-up of the LINUX server

is necessary. Appropriate documentation on the server set up are provided with the component, although to support Nurogames use-case and reduce the effort, the server was set up by the asset provider.
Eventually, we do not foresee any specific issues for third-party developers to be able to set up the server, although some smaller, less experienced studios might face time and budgeting constraints.

Game Storage - Client Side and Game Storage - Server Side

These two particular components benefit from each other and most likely will be integrated in a bundle by gaming studios. The server-side game storage that provides storage on the analytics infrastructure was easily integrated. It is rather simple and easy to implement since it provides an API-based key-value storage backend storing all the data in a JSON format. Eventually, this component serves as a storage backend for Client-side component, but for the time of the game implementation the client-side component did not provide support of C++. A general question an independent game studio or developer might raise is whether it is worth to invest time for creating a wrapper around this component or develop a simple solution from scratch (time-wise the latter might be even faster).

Server-side Dashboard and Analysis Asset

The dashboard provides analytics and an interface for the interaction storage and analytics component. It was successfully implemented and used during the WaterCooler interactions analysis. It serves as a web-based interface providing various dashboards and has a clear benefit for the stakeholders using the game, interacting with the content, teachers etc. All in all, added value to the game with low effort.

Dialogue Scenario Editor

The dialog scenario editor is designed for the domain expert who is able to develop a dialogue scenario as a graph of steps with scores and feedback related to each step. The features allow to develop conversations between the NPCs and the player with multiple choices. The output of the editor is stored in an XML format that is easy to integrate in the game. It does not require for a native client application, since it runs in a browser (JavaScript). The component is well documented providing Demos for the domain expert, running and building instructions as well as deployment instructions.

Dialogue Player / Reasoner / Step-based competency assessment

This particular asset is an added value being able to parse the output of the Dialogue Scenario Editor. The scenario parser can generate scenario data to perform dialogue management. It converts XML output to a binary representation and offers services through a web-service using JSON-RPC, making it possible to call from the games/software implemented in various languages. No issues were faced during the integration process. Nurogames has been using it as a server-side asset, but the addition of a client-side which is also provided can benefit many studios who are planning to use a combination of a scenario editor and a reasoner.

Evaluation Asset

This component captures the log data of the game and transmitted to the server for interaction/outcomes/assets evaluation. Easy to integrate and establish communication through REST API. The component is documented with the source code, manual and some relevant fact sheet.

Overall, Nurogames did not encounter any significant difficulties concerned with integration or interoperability.

3. ASSET DESCRIPTIONS AND NOMENCLATURE

This section deals with the Application Programming Interfaces (API); these include routines, protocols, and tools for building software applications.

At the time of writing all of the Assets with available API descriptions have been checked thoroughly for usage and their specific terms. The specific terms are listed in a read-only document entitled T1-2summary Available at https://docs.google.com/spreadsheets/d/1e_Mm6gcvFzaT7s39xbl-RecupwGjGXxBrc2gS7I-oxw/edit#gid=612519042

The majority of the assets use unique terms in their API descriptions and very few used shared terms. The majority of the API documents provide lists of their methods usually generated by a tool without functional explanation of their work and semantics. Consequently, for these assets no common terms can be extracted. Explanations of approaches and functioning are reported in scientific studies. No inconsistencies have been identified in the API documents and it is essential that as new assets are accepted into the Ecosystem portal that the API documents remain consistent with the Asset overview page, Asset tutorial/quick start, deployment instruction and eventually the API document for contributors.

In the RAGE project there has been very little demand for interaction between the assets from developers, the exception to this being for clusters of interoperable assets developed together. To this end, various combined assets have been made available as aggregate bundles.

The project interoperability and integration team investigated the requirement for an opportunity to develop a formal Nomenclature for the project. It was however identified that a wide variety of nomenclature were being used by developers. For example, the xAPI profile development detailed in section 4 of this document, exhibits an implied nomenclature.

Consistent with the pragmatic approach and ethos of the project, nomenclature details became more of an inspection than a development task and this was identified by both the Game and Asset developers as a positive outcome providing them with the flexibility to focus on innovation over compliance. The final outcome is that a de-facto nomenclature may emerge over time, however it does not present a problem for developers to use their own labels and descriptions.

4. SPECIFICATIONS AND STANDARDS

As detailed in section 1 of this deliverable (Integration Approach and Rationale) and Integration and Interoperability working group was established in the RAGE project. The purpose of the group was to ensure that a pragmatic approach to the use of specifications and standards was adopted by the RAGE project. Specifications or standards were adopted where they added value to the project in terms of ease of use, technical interoperability or in market development.

The group interviewed all asset and component developers to ascertain which specifications or standards were being used and provided additional advice on where they may enhance functionality or improve technical integration and interoperability. It was envisaged that a standards catalogue may be developed and made available to asset or game developers. This activity was however superseded by the production of an Asset matrix developed within Work-package 4 of the RAGE project. The matrix provided a framework for mapping all assets to their use case study games. Where gaps were identified specifications and standards were suggested (subject to the above criteria).

The standards considered in this process were drawn from the Technology Enhanced Learning (TEL) domain and included the current suite of IMS GLC specifications, International Standards Organisation (ISO), Advanced Distributed Learning (ADL) and Institute of Electrical and Electronics Engineers (IEEE). The International Standards Organisation (ISO) standards are not “open”, involving a fee for each implementation, and as such incorporation in the RAGE project was impractical and against the project objective of the implementation of “Open” standards.

The IMS GLC Specifications Considered

Accessibility
Accessibility Portable Item Protocol (APIP)
Calliper Analytics
Competency Definitions
Common Cartridge (CC)
Competencies and Academic Standards Exchange
Course Planning and Scheduling
Enterprise
E Portfolio
Learner Information (LIP)
Learning Design (LD)
Learning Tools interoperability (LTI)
Meta data
Question and Test Interoperability (QTI)
Shareable State Persistence (SSP)

The ADL Specifications Considered

Shareable Content Object Reference Model (SCORM)
Experience Application Programme Interface (xAPI)

IEEE Specifications Considered

IEEE 1484.20.1-2007 - IEEE Standard for Learning Technology-Data Model for Reusable Competency Definitions

IEEE 1484.1-2003 - IEEE Standard for Learning Technology - Learning Technology Systems Architecture (LTSA)

In the Description of Works (DOW) the ADL Shareable Content Object Reference Model (SCORM) specification was highlighted as being of potential benefit to the project. This was

considered but the decision to use the more recent ADL xAPI standard, with greater functionality and flexibility offered, was taken and this work is specifically detailed in the next section of this document.

Similarly, the application of IMS Common Cartridge (CC) was considered. This is a specification concerned with content packaging. There was no demand from either asset or game developers to adopt the specification and more significantly no demand from the Case studies, representing both education and Training organisations. Further investigation revealed little take up or interest in using the specification (CC) outside the United States and South Korea where the established publishing model is conducive to consuming packaged content within Learning Management Systems (LMS). However, it was identified that exchange of data (using xAPI), as opposed to running packaged content, within an LMS or Virtual Learning Environment (VLE) could be of significant interest in European and other global markets. This prompted the instigation of the Moodle integration work detailed and covered in depth in the next section of this deliverable.

What became evident within the project was the ongoing tension between the perceived restrictions of standards on the creative development process, largely exhibited amongst the game development partners, and the overarching RAGE project objective of using interoperable assets to stimulate Applied Games market growth.

The RAGE project working group managed this tension and the adoption of a pragmatic approach resulted in not only the application, testing and proof of concept use of technical specifications but in the project contributing to international specification initiatives

5. CONTRIBUTIONS TO SPECIFICATIONS AND STANDARDS

This section provides details of work undertaken in the RAGE project that has contributed to International formal specifications and standards bodies. This concerns the serious games profile for xAPI which was developed by the project.

5.1 The Serious Games Profile for xAPI

To provide generalizable and scalable solutions in RAGE we decided to use standards specifications specifically for the learning analytics aspects. After an analysis of the domain it was determined that Experience API was the most suitable specification in the e-learning domain that could be deployed in the applied game domain. In the original DOW the use of SCORM was detailed. Whilst not replicating the functionality of SCORM, which is now quite a mature specification, xAPI was selected as more appropriate for the project requirements.

The Experience API (xAPI) is an e-learning specification that provides for the collection of data relating to the experience of a person both on and offline in a learning environment. This API standardises the capture of data from activities from different technologies, so a number of different systems can securely communicate transferring their data in xAPI vocabulary. Each event tracked in a learning activity in xAPI format is called a statement. The specification was created by an open community led by the Advanced Distributed Learning Initiative (ADL).

The aim is that the specific profiles cover use cases across different domains. The project developed the xAPI-SG Profile for the domain of Serious Games (SGs).

5.1.1 xAPI-SG Profile

The xAPI-SG Profile defines a common set of verbs, activity types and extensions that are common for SGs, aiming that any learning activity involving a SG (i.e. any interaction with a SG) can be traced using the vocabulary defined in the xAPI-SG Profile. The Profile aims to be general enough to cover the interactions of players in most games (being game-independent) but concrete enough to provide meaningful information for the different stakeholders (e.g. students, teachers, developers, educational institutions).

5.1.2 Tracking xAPI-SG data

The main objective of the xAPI-SG Profile is to facilitate the representation and communication of data regarding learners' interactions with a game. The xAPI-SG statements collected can then be analysed for a variety of perspectives to extract useful information about the use of games. The information derived from the analysis of the xAPI-SG data can be used for different purposes (to validate game or learning design, to identify errors or areas for improvement in the game, to provide visual feedback), at different times (near real-time while students are still playing or at a later stage for batch analysis) and be useful for different stakeholders (students, teachers, game developers, educational institutions).

Details info about xAPI-SG statements can be found at https://github.com/e-ucm/rage-analytics/wiki/Experience-API-(xAPI)-Statements.

The xAPI Serious Game profile is fully operational and currently being used in a number of projects projects both in Europe (H2020 - Beaconing, Erasmus+ IMPRESS) and in the USA (collaboration with GBL-xAPI NSF initiative https://gblxapi.org/)

6. INTEGRATION WORK UNDERTAKEN

In this section we provide detail of specific Integration and Interoperability activities undertaken focussing on the proof of concept, which involved the application of the IMS GLC open specification Learning Tools Interoperability (LTI) to three different scenarios. This entails the exchange of data captured using the xAPI profile (applied within the RAGE pilot games) with an established Learning Management System (LMS) or Virtual Learning Environment (VLE).

6.1 LTI Moodle Proof of Concept for RAGE

The purpose of this piece of work was to investigate as a "proof of concept" integrating the RAGE analytics suite, in particular the analytics service, with typical education applications such as Moodle using the IMS LTI specification. The work was not originally specified within the DOI but provides an important extension of the content of this deliverable.
This development work built on the existing interoperability work undertaken in the project using the xAPI standard a specification that has been incorporated into several of the pilot games. Using the existing game asset and data of a RAGE game the proof of concept is focussed on the export of data into a Virtual Learning Environment (VLE) or Learning Management System (LMS) at the University of Bolton via a tool integration using the IMS LTI (Learning Tools Interoperability) specification. This work will be required to explore the exportation and integration of meaningful student data in the VLE in the context of security and General Data Protection Regulation (GDPR) issues that have emerged during the course of the project.
This section outlines the findings of this investigation. Three integration scenarios were investigated

- Scenario 1 The RAGE analytics server as an LTI gateway
- Scenario 2 LTI games loosely coupled to RAGE analytics
- Scenario 3 LTI games integrating RAGE analytics as a service

6.1.1 Scenario 1: RAGE Analytics server as LTI gateway

UCM developed an LTI connector into the RAGE Analytics service; in this model, the presentation of the game is via the RAGE platform, and teachers need to add RAGE as an external tool using a URL, key and secret obtained from the RAGE platform. In this scenario, the relationship between academic and game is mediated by the RAGE analytics suite - each teacher needs to register with this platform to create a class, add the game, and obtain the connection details.
However, the game itself cannot be launched from within RAGE - instead the game developer would need to create *another* LTI interface to the game itself if teachers were going to embed the game in the VLE.
This usage scenario seems best placed for viewing the statistics for non-web-based games that by necessity cannot be run within a VLE, so where games are deployed to be played on the desktop, and then the analytics viewed within the VLE.

Development
Most of the implementation involved the setup of the RAGE analytics server on Windows Azure; the current releases do not support LTI, so a pre-release version was used. It is essential that developers read the supporting documentation and follow the mandatory recommendations (OS support, RAM and CPU requirements) to avoid potential problems.
Bugs were identified in the software that were swiftly addressed and problems rectified by UCM and this has served to highlight areas for future development for third party users.

User Setup
First, as a Developer user one needs to register, login and create a game with a tracking code and game link.

As a Teacher user, one needs to register, login, and create a class and an activity. The activity is set to also allow "anonymous" users. From the Class view we then enter a shared secret and generate the LTI connection details:

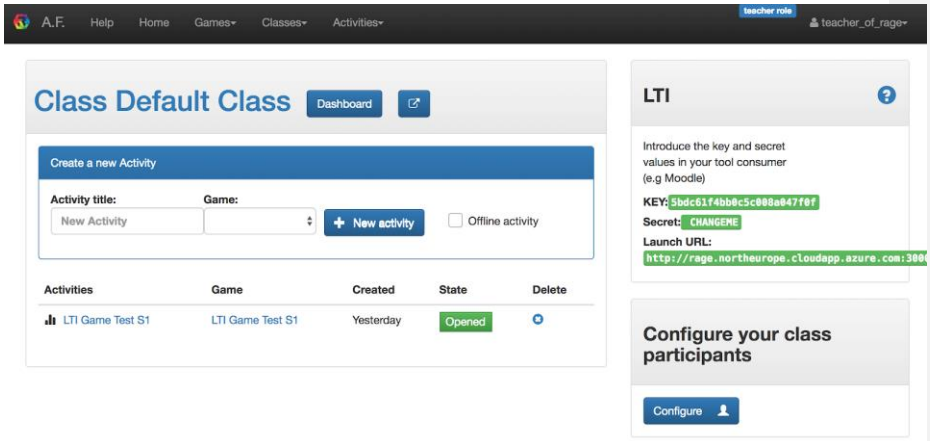

Figure 1 User Set Up Screen Shot

We then added these details to Moodle:

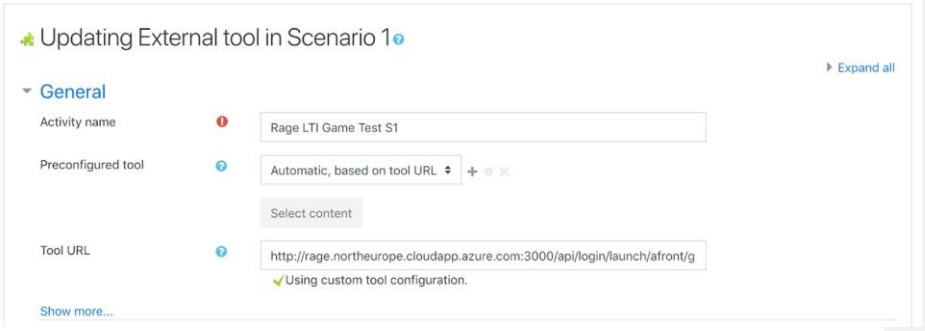

Figure 2 Updating External Tool Screen Shot

The game is then visible as an activity within Moodle:

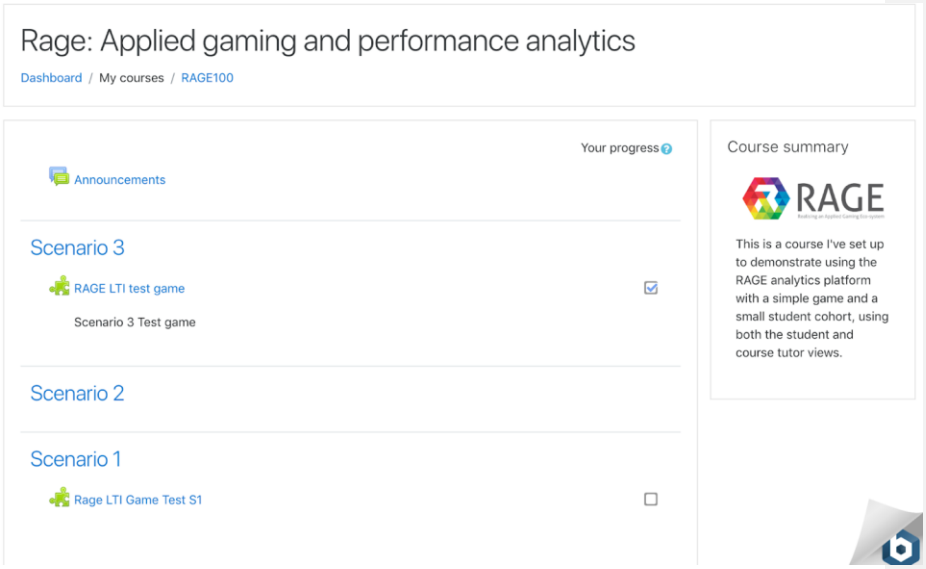

Figure 3 Course Summary Screen Shot

When a student clicks on the activity, the RAGE a2 portal is visible, and presents an interface to any analytics related to the activity:

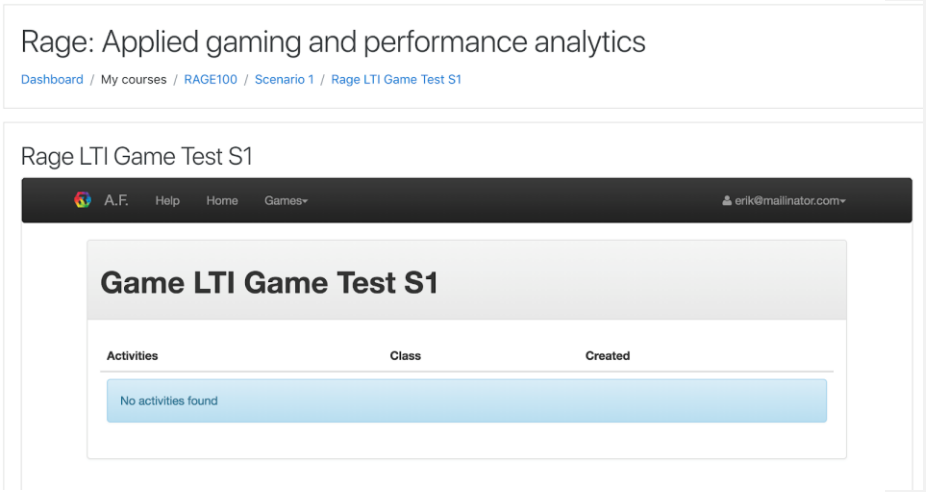

Figure 4 Scenario 1 Test Game Screen shot

The LTI launch should only be used to access analytics, not for launching the game. The game developer therefore needs to create a separate LTI implementation for their game to be launched from within Moodle.

The LTI implementation in the RAGE Analytics platform is not intended to be used for launching games in a VLE, but for launching the platform itself and enabling account setup by learners. The teacher needs to provide some instructions to students for how to launch the game separately from the analytics platform.

6.1.2 Scenario 2: LTI Games loosely coupled to RAGE Analytics

An alternative scenario is for the game itself to implement LTI, and to send traces of activity to the RAGE analytics server. Teachers are then required to separately access the RAGE analytics suite if they wish to view analytics. To some extent this is a challenging scenario from a usability perspective, as teachers need to both add the game and set up the analytics using RAGE, and also set up the game as an external tool in Moodle. However, it is also broadly similar l to Scenario 1.
Overall, this usage scenario fits where an existing LTI-based game is integrated with RAGE analytics in a fairly superficial way.

Development
A mock game was developed using Python/Flask and JavaScript. This included the PyLTI library for connecting with LTI services, and the UCM JavaScript xAPI library for sending traces to the RAGE server. The game was deployed on Windows Azure. LTI functionality was tested using Saltire. The game used the UCF LTI Flask template (https://github.com/ucfopen/lti-template-flask) and the UCM Tracker demo app (https://github.com/e-ucm/js-tracker).

User Set-Up
From the game developer perspective, setting up the activity for use by teachers requires logging into the RAGE analytics platform, adding the game, and obtaining the tracking identifier; in the demo Flask app this is added as an environment variable along with the location of the RAGE server, and the game developer's login credentials.
From the teacher perspective, they must first log in to the RAGE Analytics front end, create a class, and create an activity using the game. Participants do not need to be manually added, but the "anonymous" box needs to be checked.
The game is now ready for use; the game can be added to Moodle in the same fashion as Scenario 3, and launched from within the course. Unlike in Scenario 3, there is no "teacher view" presented by the game; instead to view analytics the teacher needs to log in to RAGE and view them there. Data is recorded using anonymous traces rather than users.

6.1.3 Scenario 3: LTI Games integrating RAGE Analytics as a service

In this scenario, the game implements LTI, and also a connector to the RAGE platform REST API. The teacher adds the launch URL, key, and secret from the game provider, but does not need to register with or know anything of the RAGE platform; the game developer instead codes the interactions with the server to set up the analytics, send the traces, and retrieve the resulting analysis. The game itself presents analysis visualisations.
The key benefit of this scenario is that the game behaves like any other external tool from the point of view of the teacher with no additional sign up required; the main drawback is the need for the game developer to perform the integration themselves.
Overall, this usage scenario fits where smaller, web-based games are being integrated into a course, for example the kind of mini games often bundled with educational content from the major academic publishers.
Limitations of the IMS Learning Tools Interoperability (LTI) Standard
In undertaking the proof of concept, the RAGE project exposed limitations of the standard. There is currently work being undertaken on version LTI1.3, however as neither of the RAGE partners are currently contributing IMS GLC contributing members access to the standard was limited to version LTI 1.2 . Identified usability issues were to be addressed in version LTI 2.0 and IMS thin Common Cartridge. Adopting the LTI 2.0 specification could potentially have resulted in an improved experience for consumers teachers and end users however work has not progressed on this new specification.

The detailed scenarios do have limitations as all employed the "anonymous login" in analytics. This was a conscious decision as addressing the requirements of GDPR was beyond the scope of the proof of concept. We recognise this as a limitation as Teachers will, of course, require data on individual performance in a fully functional implementation.
The LTI standard itself presupposes a 1:1 integration of Tool Provider and Tool consumer.
In the proposed scenarios there are 3 tools involved, Moodle as a Tool Consumer, the UOB game platform as a provider, and the analytics suite as a provider. In this scenario, there will be two 1:1 (Moodle - Game platform and Moodle - Analytics) integrations, but in addition a Game Platform - Analytics integration is required. This last integration could be a non-standard (scenario 3) or could be in done in a standard way (using LTI, with the game-platform acting as a Tool Consumer).

Development
A mock game was developed using Python/Flask and JavaScript. This included the PyLTI library for connecting with LTI services, and the UCM JavaScript xAPI library for sending traces to the RAGE server. The game was deployed on Windows Azure. LTI functionality was tested using Saltire. The game used the UCF LTI Flask template (https://github.com/ucfopen/lti-template-flask) and the UCM Tracker demo app (https://github.com/e-ucm/js-tracker). Analytics visualisations were developed using D3.js.
The most challenging part of the implementation was setting up the RAGE service entirely using API calls, as the documentation was incomplete. However, it was possible for the game to dynamically create teachers, courses, and activities and to enrol students without any intervention from teachers, and also to retrieve the resulting data from the RAGE platform using Kibana's proxy API to ElasticSearch.
The complete codebase can be found on Github: https://github.com/scottbw/ragelti

User Setup
From the game developer perspective, setting up the activity for use by teachers requires logging into the RAGE analytics platform, adding the game, and obtaining the tracking identifier; in the demo Flask app this is added as an environment variable along with the location of the RAGE server, and the game developer's login credentials.
From the teacher perspective, the process of using the game is very simple. A teacher adds the tool to Moodle in the standard way:

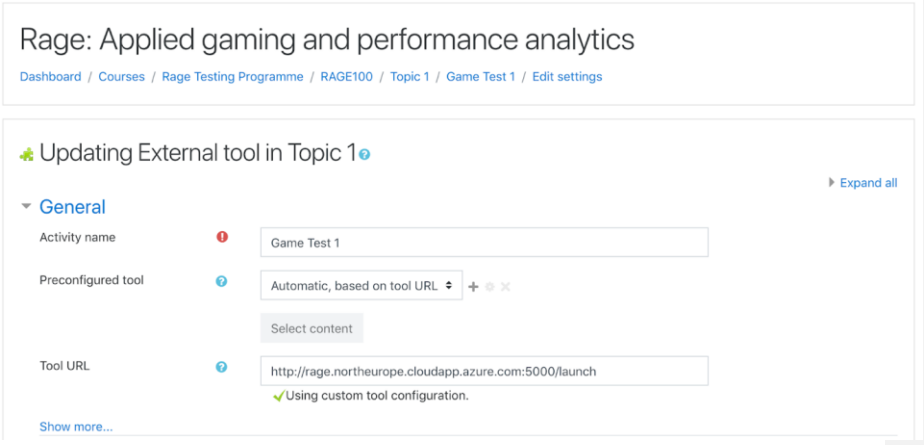

Figure 5 Updating External tool Screen Shot (1)

The game is then visible as an activity within Moodle:

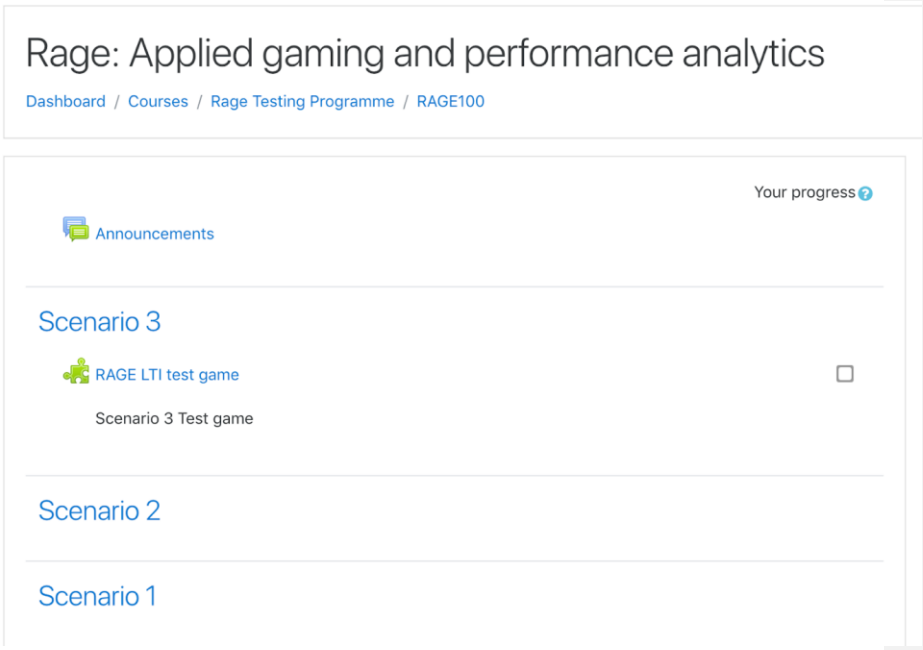

Figure 6 Applied Gaming and Performance Analytics Screen Shot

When a teacher clicks on the activity, the game launches with the teacher view, which displays some simple visualisations of data pulled from RAGE:

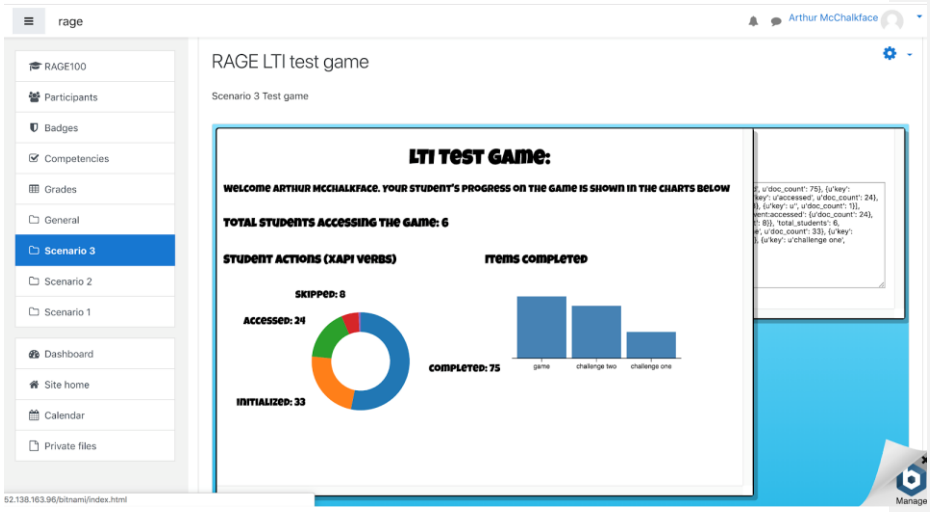

Figure 7 LTI Test Game Screen Shot (1)

When a student clicks on the activity, the game launches with the student view, which presents the game itself, and interacting with the game sends activity traces to the RAGE Analytics server:

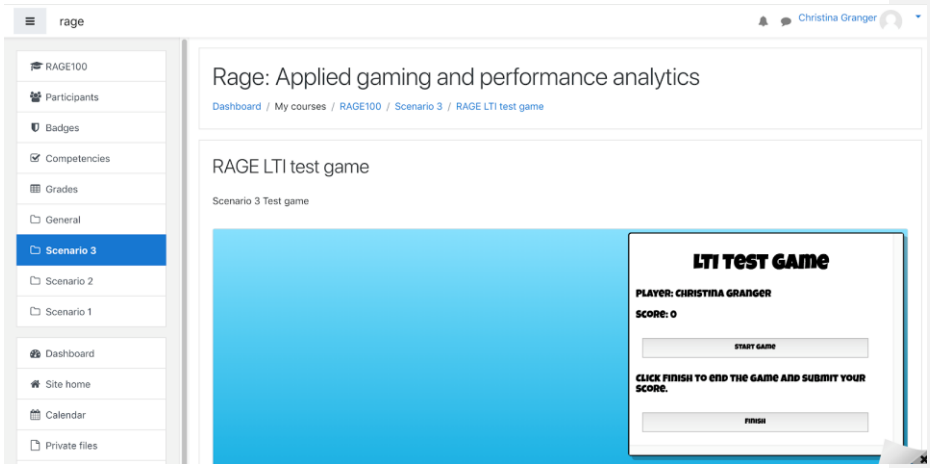

Figure 8 LTI Test Game Screen Shot (2)

The actual options presented to students to test tracing were to view or skip an intro video, to complete two optional items, as well as starting and finishing the game. Each of these actions triggers the sending of an xAPI statement to the RAGE analytics server using the JS Tracker library.
Once the game is finished, the game uses LTI to send scores to the Moodle gradebook. This turned out to be problematic using my Bitnami Moodle VM image, but worked fine when testing via Saltire:

Figure 9 Gradebook Screen Shot

6.1.4 Additional Integration and Interoperability Activity

Additional integration and interoperability development work has been undertaken by RAGE partner FTK. Though not directly related to RAGE assets the work has contributed to one of the principle aims of the project, stimulating The Applied Games Industry in Europe, in enabling gaming technologies to be seamlessly integrated with Learning Management systems (LMS).
The first integration between the Moodle Virtual Learning Environment and Virtual Reality (VR) Technology was undertaken through an LTI-Bridge. This has resulted in an interactive X3D VR being integrated with Moodle and executed at run time through an LTI call. This facilitates the exchange of Gaming results data allowing them to be mapped in the Learning Management system.

The second is an LTI bridge with the Unity engine resulting in a Unity based interactive game being integrated into a Moodle course again executed at run time through an LTI call. Whilst not directly related to asset interoperability the work will ensure that once assets are integrated into games they will not disrupt functionality or performance.

6.1.5 Findings

In undertaking the proof of concept, the RAGE project exposed limitations of the LTI standard. There is ongoing work being undertaken on version LTI1.3, however as neither of the RAGE partners are currently IMS GLC contributing members, access to the standard was limited to version LTI 1.2. Identified usability issues were to be addressed in version LTI 2.0 and IMS thin Common Cartridge, this would have resulted in an improved experience for consumers teachers and end users. However, work has not progressed on this new specification. The detailed scenarios had limitations as all employed the "anonymous login" in analytics. At this stage scenario 3 is the most practicable setup able to produce useful analytical data to inform tutors of progress through the game.

As currently configured, the LTI capability within the RAGE Analytics Platform is provided as means to access analytics about a game, where that game is accessed outside the VLE; it best fits the scenario of providing supporting analytics for desktop-based games, and requires specialised setup. This would limit potential use by a lone teacher wanting to try out a game with students given the amount of specialised knowledge required, and the time needed for configuration. It is best suited to the scenario where a teacher has the support of dedicated technicians for a high-stakes, high-value activity that form a major part of a course, or where a provider is buying in the expertise as part of a service to deliver the game.

However, as we were able to demonstrate, the underlying APIs provided by the RAGE platform make it possible to create an excellent, seamless experience for teachers and students for playing web-based games and visualising analytics, all within the VLE. This does involve effort from the game developer in the initial setup, but the amount of work is both reasonable and likely within the developer's skill set.

For future consideration for development could be a Rage Analytics SDK plugin that makes the main tasks of initialising an activity for a given LTI launch simpler, and perhaps also include some of the basic analytics queries.

In summary, it's important to communicate clearly to potential users of the RAGE assets the different approaches to integration and interoperability, and the potential resources and skills required to deliver them, to set reasonable expectations. This information can in the future be provided to potential developers through the RAGE portal.

7. CONCLUSIONS

The project decision to adopt a pragmatic approach to standards and specifications and where possible use open standards has been validated with developers to uptake current and evolving specifications free of any requirement of conformity to a formal standards framework or policy.

The decision has resulted in both Asset and Game developers being fully focussed on research and development of their outputs at the “cutting edge” as opposed to compliance with formal standards. It has also allowed for flexibility and served to highlight potential areas for improvement, future development.

As a consequence of the RAGE project approach to integration and interoperability the project has been able to contribute significantly to international specification and standards development in the form of the xAPI-SG profile developed by project partner UCM.

The requirement of a flexible approach to descriptions of assets resulted in a review of the projects original intent in “developing” a formal standardised nomenclature. Work with developers confirmed that less formal or an implied nomenclature was preferable, thereby providing the developers with flexibility. As highlighted in the document, an observed nomenclature may indeed develop as practice with Asset based development in Applied Games becomes more established.

The work package activities have highlighted areas for future development through exposing the assets and games to rigorous user testing with notable and expected improvements between the pilot phases one and two.

The project has also undertaken the first significant proof of concept work in integrating the outputs of Applied games into a Virtual learning Environment (VLE) serving to test both the functionality and robustness of the RAGE analytics suite and of the Learning Tools Interoperability (LTI) specification used to bridge the software and platform with further work planned in this domain.

Other challenges were highlighted in the reflections of the RAGE game development partners in terms of documentation and functionality that may be resolved in the future, but have served to inform developers of pragmatic considerations of implementation for example the challenges of working with corporate firewalls and security controls

None of the Game development partners highlighted any **significant** problems in integrating the RAGE Assets, which is a positive outcome for the project and validates both the RAGE Asset development concept and the approach taken to integration and interoperability.

8. REFERENCES

https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/78889/Government-Response.pdf (Accessed 2111/18)

www.imsglobal.org (Accessed 21/11/18)

https://adlnet.gov/ (Accessed 27/11/18)

https://gblxapi.org/) (Accessed 21/11/18)

https://standards.ieee.org/ (Accessed 27/11/18)

Ángel Serrano-Laguna, Iván Martínez-Ortiz, Jason Haag, Damon Regan, Andy Johnson, Baltasar Fernández-Manjón (2017): *Applying standards to systematize learning analytics in serious games*. Computer Standards & Interfaces 50 (2017) 116–123, http://dx.doi.org/10.1016/j.csi.2016.09.014

Cronbach, L. J. (1951): Coefficient alpha and the internal structure of tests. In: Psychometrika, 16, 297-334 (28,307 citations in Google Scholar as of 4/1/2016).

Davis, F. D.:(1989) Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. In: MIS Quarterly 13(3): 319-340 .

Siegmund, J., et al.(2014): Measuring and modeling programming experience. In: Empirical Software Engineering 19(5): 1299-1334 .

Van der Vegt, W., Nyamsuren, E., & Westera, W. (2016, 6-7 June). RAGE Reusable Game Software Components and Their Integration into Serious Game Engines. In G. M. Kapitsaki & E. Santana de Almeida (Eds.), Bridging with Social-Awareness, 15th International Conference, ICSR 2016, Proceedings, Lecture Notes in Computer Science, Volume 9679 2016 (pp. 165-180). Limassol, Cyprus.

Van der Vegt,W. Westera,W Nyamsuren,E. Georgiev,a. and Martinez Ortiz,i.(2016): RAGE architecture for reusable serious gaming technology components. In: International Journal of Computer Games Technology. Article ID 5680526. DOI: 10.1155/2016/5680526.

Van der Vegt,W. Bahreini,K. Nyamsuren,E. and Westera,W.(2018): Toward reusable game technologies: assessing the usability of the RAGE component-based architecture framework (under submission).

Field Code Changed